Narzędzia informatyczne w językoznawstwie

Perl - Wprowadzenie

Marcin Junczys-Dowmunt
junczys@amu.edu.pl

Zakład Logiki Stosowanej
http://www.logic.amu.edu.pl

21. listopada 2007

Co to jest Perl?

- Perl jest dynamicznym językiem programowania stworzonym przez Larry'ego Walla (lingwistę) w roku 1987
- Perl jest językiem bardzo wysokiego poziomu (C – niskiego poziomu, Java – wysokiego poziomu)
- Perl jest językiem skryptowym, interpretowanym (oraz kompilowanym)
- Perl jest językiem proceduralnym, obiektowym lub/i funkcjonalnym (w porównaniu: Pascal – proceduralny, Java – obiektowy, Haskell – funkcjonalny, Prolog – deklaratywny)
- Perl jest językiem wielozadaniowym ze szczególnym naciskiem na przetwarzanie dużych ilości danych w postaci tekstu

Krótka historia Perl

- Larry Wall tworzy pierwszą wersję w grudniu 1987
- W roku 1988 powstaje wersja Perl 2, w 1989 Perl 3
- W roku 1991 pojawia się książka *Programming Perl* autorstwa Larry'ego Walla (tzw. Wielbłąd – the Camel Book). Numer wersji zostaje podniesiony do 4, żeby zasygnalizować zgodność z tym dziełem
- W roku 1994 pojawia się Perl 5, który jest stale rozszerzany. Najbardziej aktualną wersją jest Perl 5.8.8
- W roku 1995 powstaje CPAN – Comprehensive Perl Archive Network – zbiór rozszerzeń, który dzisiaj zawiera 11,000 modułów pochodzących od ponad 5000 autorów.
- Obecnie trwają pracę nad zupełnie nową wersją Perl 6

Dlaczego właśnie Perl?

- Perl powstał jako język służący do przetwarzania dużych zbiorów tekstu
  - Pełna integracja wyrażeń regularnych
  - Wbudowana i prosta obsługa dynamicznych struktur danych: list, tablic i tablic asocjacyjnych (tzw. *hashe*) oraz struktur bardziej złożonych
  - Prosta obsługa operacji wejścia/wyjścia w przypadku wczytywania plików tekstowych
- zwięzła składnia – średnio jeden wiersz kodu Perla to tyle co 6 wierszy C++
- Gigantyczna biblioteka ogólnodostępnych modułów (CPAN - Comprehensive Perl Archive Network) np. parsery XML, HTML, obsługa Unicode itp.
- Od lingwistów pracujących w przemyśle informatycznym najczęściej wymaga się znajomość Perla

Perl jako zjawisko kulturowe

- Wokół Perla powstało coś w rodzaju subkultury zwolenników tego języka (dzieje się podobnie w przypadku innych języków skryptowych)
- Motto: "There is more than one way to do it" – skrót: `TIMTOWTDI` wymawiane jako *Tim Toady*
- Ksywy:
  - Practical Extraction and Report Language
  - Pathologically Eclectic Rubbish Lister
  - The Duct Tape of the Internet
  - The Swiss Army Chain Saw of Programming Languages
- "Obfuscated Perl Contest" – konkurs na pisanie najbardziej nieczytelnego programu, który nadal "coś" robi
- Zawody poezji pisanej w Perlu, która nadal jest poprawnym programem (tzn. kompiluje się)

Poezja w Perlu - przykład

Laureat pierwszego konkursu (2000) poezji pisanej w Perlu:

```
1       if ((light eq dark) && (dark eq light)
           && ($blaze_of_night{moon} == black_hole)
           && ($ravens_wing{bright} == $tin{bright})){
           my $love = $you = $sin{darkness} + 1;
5       };
```

Na podstawie wiersza "The Invocation" Jim'a Steinman'a

If light were dark and dark were light
The moon a black hole in the blaze of night
A raven's wing as bright as tin
Then you, my love, would be darker than sin.

Alternatywy

- Obecnie najważniejszą alternatywą do Perla jest język skryptowy Python, któremu ostatnio mocno po piętach depcze język Ruby
- Cała trójka różni się między sobą raczej filozofią niż funkcjonalnością
- Jak dotąd Perl jest najpopularniejszym wielozadaniowym językiem skryptowym w internecie [1]

Więc dlaczego właśnie Perl?

Ponieważ prowadzący nie zna ani Phytona ani Ruby (podejrzewa jednak, że znajomość jednego języka z tej trójki jest w zupełności wystarczająca)

[1] omijamy tutaj PHP, który jest językiem przeznaczonym praktycznie tylko i wyłącznie do tworzenia stron internetowych

Dystrybucje Perla

- **ActivePerl** – Najpopularniejsza dystrybucja Perla dla Windowsa autorstwa Activision – łatwy dostęp do (nie do końca aktualnej wersji) CPAN poprzez PPM (Perl Package Manager)
  http://www.activestate.com/store/activeperl/download/
- **Cygwin** – Nie tyle dystrybucja Perla, raczej coś w rodzaju symulatora Linuxa pod Windowsem, zawiera wersję Perla
  http://www.cygwin.com/
- Istnieje wiele innych dystrybucji, ale wydają się one nie do końca dopracowane

Jak tworzymy program w Perl?

- Programy Perl (skrypty) to zwykłe pliki tekstowe (najczęściej z rozszerzeniem `.pl`)
- Możemy je edytować dowolnym edytorem tekstu, jednak polecam edytory z podświetlaniem składni (np. EmEditor) lub środowiska bardziej wyspecjalizowane (np. Activision Komodo Edit – darmowa wersja Komodo IDE)
- Skrypty wykonujemy z poziomu wiersza poleceń, wpisując `perl nazwa_skryptu.pl`

Zliczanie wyrazów - wersja czytelna

```
1  use strict;

   my %conc;
   while (my $line = <>) {
5      chomp $line;
       my @words = split /[ .\?,!;:\(\)]+/, $line;
       foreach my $word (@words) {
           if (exists $conc{$word}) {
               $conc{$word}++;
10         }
           else {
               $conc{$word} = 1;
           }
       }
15 }
   foreach my $word (sort keys %conc) {
       print "$word: $conc{$word}\n";
   }
```


Zliczanie wyrazów - wersja hakerska

```
1  while (<>) {
       chomp;
       $conc{$_}++ foreach (split /[ \.\?,!;:\(\)]+/);
   }
5  print "$_: $conc{$_}\n" foreach (sort keys %conc);
```

Skrócenie powstało przez:

- Korzystanie ze zmiennych domyślnych
- Wykorzystanie faktu, że Perl potrafi automatycznie deklarować i definiować zmienne
- Zapis skrótowy pętli, jeśli wykonany jest tylko jedno polecenie
- Inne mechanizmy: np. interpolacja zmiennych

Ekonomia zapisu – wada czy zaleta?

- Istnieje zarzut, że taki styl programowania promuje pisanie nieczytelnego kodu
- To prawda – jednak napisania 4 wierszy kodu zajmuje o wiele mniej czasu niż napisania 17 wierszy
- Zaleta: Perl pozwala na pisanie programów jednym i drugim stylem: czytelnym i czystym lub szybkim i brudnym.
- Własne doświadczenie: im lepiej znamy Perla, tym chętniej korzystamy ze stylu skróconego
- Wspomniane zawody "Obfuscated Perl"

Nasz pierwszy program

Utwórzmy plik tekstowy `hello.pl` zawierający następujący wiersz:

```
print("Hello, world!\n");
```

- Typowy pierwszy przykład w (prawie) każdym podręczniku programowania
- Wykorzystamy go do zabawy z komunikatami o błędach
- Dowiemy się w ten sposób jakie polecenia są według Perla "gramatyczne" a jakie nie

Najpierw wykonamy nasz program z poziomu wiersza poleceń komendą (o ile znajdujemy się w tym samym katalogu)

```
perl hello.pl
```


Anatomia programu

```
print("Hello, world!\n");
```

- `print` jest **funkcją**, czymś w rodzaju czasownika
- Fragmenty ujęte w nawiasie to **argumenty** funkcji – inaczej dopełnienia czasownika
- Wszytko w cudzysłowie to **łańcuchy znakowy**, możemy je traktować jak nazwy lub imiona
- całość zakończoną średnikiem nazywamy **instrukcją** (zdania w języku naturalnym)
- Nasz prosty program przykładowy składa się tylko z jednej instrukcji

Komunikaty o błędach

Ćwiczenie

- Proszę zmodyfikować program `hello.pl`
- możemy np. :
  - powtórzyć wiersz programu, usunąć średniki w różnych kolejnościach
  - usunąć nawiasy
  - dodać białe znaki (spacje, taby, załamania wiersza) w dowolnych miejscach programu
  - zmienić `print` na jakąś wersję ortograficznie niepoprawną
  - usunąć znaki cudzysłowu
  - zmienić znaki cudzysłowu podwójnego na cudzysłów pojedynczy
- Po każdym kroku wykonać program `hello.pl` i przeanalizować komunikaty błędu (o ile się pojawią) i wyświetlaną treść

Komunikaty o błędach

Teraz zmodyfikujemy nasz mały program do następującej postaci:

```
use diagnostics;
print "Hello, world!\n";
```

Powtórzymy poprzednie modyfikacje i przeanalizujemy ponownie komunikaty o błędach. Tym razem będą one znacznie bardziej wyczerpujące.

Komunikaty zostaną wyświetlone na wyjściu błędów. Żeby móc je obejrzeć ekran po ekranie, trzeba je przekierować na wyjście standardowe:

```
perl hello.pl 2>&1 | more
```

Komunikaty o błędach

Wnioski

- Nawiasy nie są konieczne przy wywołaniu funkcji (o ile nie ma wieloznaczności), ale zwiększają czytelność
- Białe znaki nie mają wpływu na program, o ile nie pojawiają się w cudzysłowie
- Cudzysłów (podwójny lub pojedynczy) jest konieczny do oznaczania łańcuchów znakowych
- Ostatni średnik (wewnątrz bloku) możemy opuścić, opuszczenie pozostałych spowoduje wystąpienie błędu
- Nasz program nie wykona się, jeśli popełniliśmy błąd składniowy
- Ważna zasada: najpierw poprawiamy pierwsze pojawiające się błędy. Pozostałe mogą być wtórne

Literatura

- Michael Hammond, *Programming for linguists: Perl for language researchers*. Oxford: Blackwell, 2003
- Randal L. Schwartz, Tom Phoenix, Brian d'Foy, *Perl. Wprowadzenie. Wydanie IV*. O'Reilly/Helion, 2006
- Larry Wall, Tom Christiansen, Jon Orwant, *Programming Perl. Third Edition*. O'Reilly, 2000
- Tom Christiansen, Nathan Torkington *Perl Cookbook, Second Edition*. O'Reilly, 2003
- Simon Cozen *Perl. Zaawansowane programowanie. Wydanie II*. O'Reilly/Helion, 2006